Class 17: Structural Induction

Schedule

Problem Set 7 is due **tomorrow at 6:29pm**.

Next week Friday, 4 November at 11am, **Steve Huffman** (BSCS 2005, co-founder and CEO of Reddit) will give a Computer Science Distinguished Alumni talk in the Rotunda. If you would like to meet with Steve (either at a lunch after the talk or a meeting with students later that afternoon), send me an email with a good reason why you should be invited (only a very limited number of spaces available).

Lists

Definition. A *list* is an ordered sequence of objects. A list is either the empty list (λ), or the result of prepend(e, l) for some object e and list l.

$$\begin{aligned}
\textit{first}(\text{prepend}(e, l)) &= e\\
\textit{rest}(\text{prepend}(e, l)) &= l\\
\textit{empty}(\text{prepend}(e, l)) &= \textbf{False}\\
\textit{empty}(\textbf{null}) &= \textbf{True}
\end{aligned}$$

Definition. The *length* of a list, p, is:

$$\begin{cases} 0 & \text{if } p \text{ is } \textbf{null}\\ \textit{length}(q) + 1 & \text{otherwise } p = \textit{prepend}(e, q) \text{ for some object } e \text{ and some list } q \end{cases}$$

```python
def list_length(l):
    if list_empty(l):
        return 0
    else:
        return 1 + list_length(list_rest(l))
```

Prove: for all lists, p, `list_length(p)` returns the length of the list p.

Concatenation

Definition. The *concatenation* of two lists, $p = (p_1, p_2, \cdots, p_n)$ and $q = (q_1, q_2, \cdots, q_m)$ is

$$(p_1, p_2, \cdots, p_n, q_1, q_2, \cdots, q_m).$$

Provide a *constructuve* definition of *concatenation.*

Prove. For any two lists, p and q, $\text{length}(p + q) = \text{length}(p) + \text{length}(q)$.

Induction Summary

	Regular Induction	**Invariant Principle**	**Structural Induction**
Works on:	natural numbers	state machines	data types
To prove $P(\cdot)$	*for all natural numbers*	*for all reachable states*	*for all data type objects*
Prove **base case(s)**	$P(0)$	$P(q_0)$	$P(\text{base object(s)})$
and **inductive step**	$\forall m \in \mathbb{N}.$	$\forall (q, r) \in G.$	$\forall s \in \mathit{Type}.$
	$P(m) \implies P(m+1)$	$P(q) \implies P(r)$	$P(s) \implies P(t)$
			$\forall t$ constructable from s

Challenge-Response Protocols

1. **Verifier:** picks random challenge, y.
2. **Prover:** proves knowledge of x by revealing $f(x, y)$.
3. **Verifer:** can verify prover knows x from response, but learns nothing (useful) about x.

How can you know the website you are sending your password to is what you think it is?

